Imp questions

15 September 2025 22:06

C Programming Important Questions with Solutions

1. Basics & Data Types

Q1: Size of Data Types

```
#include <stdio.h>
int main() {
   printf("Size of int: %d bytes\n", sizeof(int));
   printf("Size of float: %d bytes\n", sizeof(float));
   printf("Size of char: %d bytes\n", sizeof(char));
   printf("Size of double: %d bytes\n", sizeof(double));
   return 0;
}
```

Output: int=4, float=4, char=1, double=8 bytes

Q2: Type Casting

```
#include <stdio.h>
int main() {
   int a = 10, b = 3;
   float result1 = a / b;          // Integer division
   float result2 = (float)a / b;    // Type casting

   printf("Without casting: %.2f\n", result1);  // 3.00
   printf("With casting: %.2f\n", result2);     // 3.33
   return 0;
}
```

Q3: ASCII Values

```
#include <stdio.h>
int main() {
   char ch = 'A';
   printf("Character: %c, ASCII: %d\n", ch, ch);
   printf("Next character: %c\n", ch + 1);  // B

   // Check uppercase or lowercase
   ch = 'a';
   if(ch >= 'a' && ch <= 'z')
      printf("%c is lowercase\n", ch);
   return 0;
}
```

2. Operators

Q4: Pre vs Post Increment

```
#include <stdio.h>
int main() {
   int a = 5, b = 5;

   printf("Post-increment: %d\n", a++);  // 5 (prints then increments)
   printf("Value of a now: %d\n", a);    // 6

   printf("Pre-increment: %d\n", ++b);   // 6 (increments then prints)
   printf("Value of b now: %d\n", b);    // 6
```

```
    return 0;
}
```

Q5: Ternary Operator (Find Max)

```
#include <stdio.h>
int main() {
    int a = 10, b = 20, max;
    max = (a > b) ? a : b;
    printf("Maximum: %d\n", max);  // 20
    return 0;
}
```

Q6: Bitwise Operators

```
#include <stdio.h>
int main() {
    int a = 5, b = 3;  // 5=0101, 3=0011

    printf("a & b = %d\n", a & b);   // 1 (0001)
    printf("a | b = %d\n", a | b);   // 7 (0111)
    printf("a ^ b = %d\n", a ^ b);   // 6 (0110)
    printf("~a = %d\n", ~a);        // -6
    printf("a << 1 = %d\n", a << 1); // 10 (left shift)
    printf("a >> 1 = %d\n", a >> 1); // 2 (right shift)
    return 0;
}
```

3. Control Structures

Q7: Pattern - Right Triangle

```
#include <stdio.h>
int main() {
    int n = 5;
    for(int i = 1; i <= n; i++) {
        for(int j = 1; j <= i; j++) {
            printf("* ");
        }
        printf("\n");
    }
    return 0;
}
```

Output:

```
*
* *
* * *
* * * *
* * * * *
```

Q8: Pattern - Pyramid

```
#include <stdio.h>
int main() {
    int n = 5;
    for(int i = 1; i <= n; i++) {
        // Print spaces
        for(int j = 1; j <= n - i; j++) {
            printf(" ");
        }
        // Print stars
        for(int k = 1; k <= 2*i - 1; k++) {
```

```
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

Output:

```
    *
   ***
  *****
 *******
*********
```

Q9: Floyd's Triangle

```
#include <stdio.h>
int main() {
    int n = 5, num = 1;
    for(int i = 1; i <= n; i++) {
        for(int j = 1; j <= i; j++) {
            printf("%d ", num++);
        }
        printf("\n");
    }
    return 0;
}
```

Output:

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

Q10: Break vs Continue

```
#include <stdio.h>
int main() {
    printf("Using break:\n");
    for(int i = 1; i <= 10; i++) {
        if(i == 5) break;  // Exits loop
        printf("%d ", i);
    }

    printf("\n\nUsing continue:\n");
    for(int i = 1; i <= 10; i++) {
        if(i == 5) continue;  // Skips only i=5
        printf("%d ", i);
    }
    return 0;
}
```

4. Arrays

Q15: Find Maximum and Minimum

```
#include <stdio.h>
int main() {
    int arr[] = {34, 12, 67, 23, 89, 45};
    int n = 6;
    int max = arr[0], min = arr[0];
```

```
    for(int i = 1; i < n; i++) {
        if(arr[i] > max) max = arr[i];
        if(arr[i] < min) min = arr[i];
    }

    printf("Maximum: %d\n", max);
    printf("Minimum: %d\n", min);
    return 0;
}
```

Q16: Reverse an Array

```
#include <stdio.h>
int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int n = 5;

    printf("Original: ");
    for(int i = 0; i < n; i++) printf("%d ", arr[i]);

    // Reverse
    for(int i = 0; i < n/2; i++) {
        int temp = arr[i];
        arr[i] = arr[n-1-i];
        arr[n-1-i] = temp;
    }

    printf("\nReversed: ");
    for(int i = 0; i < n; i++) printf("%d ", arr[i]);
    return 0;
}
```

Q17: Linear Search

```
#include <stdio.h>
int main() {
    int arr[] = {10, 20, 30, 40, 50};
    int n = 5, key = 30, found = 0;

    for(int i = 0; i < n; i++) {
        if(arr[i] == key) {
            printf("Element found at index %d\n", i);
            found = 1;
            break;
        }
    }

    if(!found) printf("Element not found\n");
    return 0;
}
```

Q18: Binary Search

```
#include <stdio.h>
int main() {
    int arr[] = {10, 20, 30, 40, 50};  // Must be sorted
    int n = 5, key = 30;
    int low = 0, high = n - 1, found = 0;

    while(low <= high) {
        int mid = (low + high) / 2;
```

```
        if(arr[mid] == key) {
            printf("Element found at index %d\n", mid);
            found = 1;
            break;
        }
        else if(arr[mid] < key)
            low = mid + 1;
        else
            high = mid - 1;
    }

    if(!found) printf("Element not found\n");
    return 0;
}
```

Q19: Bubble Sort

```
#include <stdio.h>
int main() {
    int arr[] = {64, 34, 25, 12, 22};
    int n = 5;

    printf("Original: ");
    for(int i = 0; i < n; i++) printf("%d ", arr[i]);

    // Bubble sort
    for(int i = 0; i < n-1; i++) {
        for(int j = 0; j < n-i-1; j++) {
            if(arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }

    printf("\nSorted: ");
    for(int i = 0; i < n; i++) printf("%d ", arr[i]);
    return 0;
}
```

Q20: Selection Sort

```
#include <stdio.h>
int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = 5;

    for(int i = 0; i < n-1; i++) {
        int min_idx = i;
        for(int j = i+1; j < n; j++) {
            if(arr[j] < arr[min_idx])
                min_idx = j;
        }
        // Swap
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
```

```
    printf("Sorted: ");
    for(int i = 0; i < n; i++) printf("%d ", arr[i]);
    return 0;
}
```

Q21: Matrix Addition

```
#include <stdio.h>
int main() {
    int a[2][2] = {{1, 2}, {3, 4}};
    int b[2][2] = {{5, 6}, {7, 8}};
    int sum[2][2];

    for(int i = 0; i < 2; i++) {
        for(int j = 0; j < 2; j++) {
            sum[i][j] = a[i][j] + b[i][j];
        }
    }

    printf("Sum matrix:\n");
    for(int i = 0; i < 2; i++) {
        for(int j = 0; j < 2; j++) {
            printf("%d ", sum[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Q22: Matrix Multiplication

```
#include <stdio.h>
int main() {
    int a[2][2] = {{1, 2}, {3, 4}};
    int b[2][2] = {{5, 6}, {7, 8}};
    int product[2][2] = {0};

    for(int i = 0; i < 2; i++) {
        for(int j = 0; j < 2; j++) {
            for(int k = 0; k < 2; k++) {
                product[i][j] += a[i][k] * b[k][j];
            }
        }
    }

    printf("Product matrix:\n");
    for(int i = 0; i < 2; i++) {
        for(int j = 0; j < 2; j++) {
            printf("%d ", product[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Q23: Transpose of Matrix

```
#include <stdio.h>
int main() {
    int matrix[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int transpose[3][3];
```

```
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }

    printf("Transpose:\n");
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

6. Strings

Q28: String Length (Without strlen)

```
#include <stdio.h>
int main() {
    char str[] = "Hello";
    int length = 0;

    while(str[length] != '\0') {
        length++;
    }

    printf("Length: %d\n", length);
    return 0;
}
```

Q29: String Copy (Without strcpy)

```
#include <stdio.h>
int main() {
    char source[] = "Hello";
    char dest[20];
    int i = 0;

    while(source[i] != '\0') {
        dest[i] = source[i];
        i++;
    }
    dest[i] = '\0';

    printf("Source: %s\n", source);
    printf("Destination: %s\n", dest);
    return 0;
}
```

Q30: String Concatenation (Without strcat)

```
#include <stdio.h>
int main() {
    char str1[50] = "Hello ";
    char str2[] = "World";
    int i = 0, j = 0;

    // Find end of str1
```

```
    while(str1[i] != '\0') {
        i++;
    }

    // Copy str2 to end of str1
    while(str2[j] != '\0') {
        str1[i] = str2[j];
        i++;
        j++;
    }
    str1[i] = '\0';

    printf("Result: %s\n", str1);
    return 0;
}
```

Q31: String Compare (Without strcmp)

```
#include <stdio.h>
int main() {
    char str1[] = "Hello";
    char str2[] = "Hello";
    int i = 0, flag = 0;

    while(str1[i] != '\0' || str2[i] != '\0') {
        if(str1[i] != str2[i]) {
            flag = 1;
            break;
        }
        i++;
    }

    if(flag == 0)
        printf("Strings are equal\n");
    else
        printf("Strings are not equal\n");
    return 0;
}
```

Q32: String Reverse

```
#include <stdio.h>
#include <string.h>
int main() {
    char str[] = "Hello";
    int n = strlen(str);

    printf("Original: %s\n", str);

    for(int i = 0; i < n/2; i++) {
        char temp = str[i];
        str[i] = str[n-1-i];
        str[n-1-i] = temp;
    }

    printf("Reversed: %s\n", str);
    return 0;
}
```

Q33: Check Palindrome String

```
#include <stdio.h>
```

```
#include <string.h>
int main() {
   char str[] = "madam";
   int n = strlen(str);
   int flag = 1;

   for(int i = 0; i < n/2; i++) {
      if(str[i] != str[n-1-i]) {
         flag = 0;
         break;
      }
   }

   if(flag)
      printf("%s is a palindrome\n", str);
   else
      printf("%s is not a palindrome\n", str);
   return 0;
}
```

Q34: Count Vowels and Consonants

```
#include <stdio.h>
#include <string.h>
int main() {
   char str[] = "Hello World";
   int vowels = 0, consonants = 0;

   for(int i = 0; str[i] != '\0'; i++) {
      char ch = str[i];
      if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')) {
         if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u'||
           ch=='A'||ch=='E'||ch=='I'||ch=='O'||ch=='U')
            vowels++;
         else
            consonants++;
      }
   }

   printf("Vowels: %d\n", vowels);
   printf("Consonants: %d\n", consonants);
   return 0;
}
```

Q35: Convert to Uppercase

```
#include <stdio.h>
int main() {
   char str[] = "hello world";

   printf("Original: %s\n", str);

   for(int i = 0; str[i] != '\0'; i++) {
      if(str[i] >= 'a' && str[i] <= 'z') {
         str[i] = str[i] - 32;  // ASCII difference
      }
   }

   printf("Uppercase: %s\n", str);
   return 0;
```

```
}
```

Q36: Count Words in String

```
#include <stdio.h>
int main() {
   char str[] = "Hello World from C";
   int words = 1;  // At least one word if string is not empty

   for(int i = 0; str[i] != '\0'; i++) {
      if(str[i] == ' ' && str[i+1] != ' ' && str[i+1] != '\0') {
         words++;
      }
   }

   printf("Number of words: %d\n", words);
   return 0;
}
```

Q37: Remove Spaces from String

```
#include <stdio.h>
int main() {
   char str[] = "Hello World from C";
   int i = 0, j = 0;

   printf("Original: %s\n", str);

   while(str[i] != '\0') {
      if(str[i] != ' ') {
         str[j] = str[i];
         j++;
      }
      i++;
   }
   str[j] = '\0';

   printf("Without spaces: %s\n", str);
   return 0;
}
```

Q38: Check Anagram

```
#include <stdio.h>
#include <string.h>
int main() {
   char str1[] = "listen";
   char str2[] = "silent";
   int freq1[26] = {0}, freq2[26] = {0};

   if(strlen(str1) != strlen(str2)) {
      printf("Not anagrams\n");
      return 0;
   }

   for(int i = 0; str1[i] != '\0'; i++) {
      freq1[str1[i] - 'a']++;
      freq2[str2[i] - 'a']++;
   }

   int flag = 1;
   for(int i = 0; i < 26; i++) {
```

```
        if(freq1[i] != freq2[i]) {
            flag = 0;
            break;
        }
    }

    if(flag)
        printf("Anagrams\n");
    else
        printf("Not anagrams\n");
    return 0;
}
```

8. Number Programs

Q39: Prime Number Check

```
#include <stdio.h>
int main() {
    int n = 29, flag = 0;

    if(n <= 1) {
        printf("Not prime\n");
        return 0;
    }

    for(int i = 2; i <= n/2; i++) {
        if(n % i == 0) {
            flag = 1;
            break;
        }
    }

    if(flag == 0)
        printf("%d is prime\n", n);
    else
        printf("%d is not prime\n", n);
    return 0;
}
```

Q40: Armstrong Number

```
#include <stdio.h>
#include <math.h>
int main() {
    int num = 153, original, remainder, result = 0, n = 0;
    original = num;

    // Count digits
    while(original != 0) {
        original /= 10;
        n++;
    }

    original = num;

    // Calculate sum of power of digits
    while(original != 0) {
        remainder = original % 10;
```

```
        result += pow(remainder, n);
        original /= 10;
    }

    if(result == num)
        printf("%d is an Armstrong number\n", num);
    else
        printf("%d is not an Armstrong number\n", num);
    return 0;
}
```

Q41: Palindrome Number

```
#include <stdio.h>
int main() {
    int num = 121, original, reversed = 0, remainder;
    original = num;

    while(num != 0) {
        remainder = num % 10;
        reversed = reversed * 10 + remainder;
        num /= 10;
    }

    if(original == reversed)
        printf("%d is a palindrome\n", original);
    else
        printf("%d is not a palindrome\n", original);
    return 0;
}
```

Q42: GCD (HCF)

```
#include <stdio.h>
int main() {
    int a = 56, b = 98, gcd;

    for(int i = 1; i <= a && i <= b; i++) {
        if(a % i == 0 && b % i == 0)
            gcd = i;
    }

    printf("GCD of %d and %d is %d\n", a, b, gcd);
    return 0;
}
```

Q43: LCM

```
#include <stdio.h>
int main() {
    int a = 12, b = 18, max;
    max = (a > b) ? a : b;

    while(1) {
        if(max % a == 0 && max % b == 0) {
            printf("LCM of %d and %d is %d\n", a, b, max);
            break;
        }
        max++;
    }
    return 0;
}
```

Q44: Sum of Digits

```
#include <stdio.h>
int main() {
   int num = 12345, sum = 0, digit;

   while(num != 0) {
      digit = num % 10;
      sum += digit;
      num /= 10;
   }

   printf("Sum of digits: %d\n", sum);
   return 0;
}
```

Q45: Reverse a Number

```
#include <stdio.h>
int main() {
   int num = 12345, reversed = 0, remainder;

   printf("Original: %d\n", num);

   while(num != 0) {
      remainder = num % 10;
      reversed = reversed * 10 + remainder;
      num /= 10;
   }

   printf("Reversed: %d\n", reversed);
   return 0;
}
```

Important Output Questions

Q46: Predict Output

```
#include <stdio.h>
int main() {
   int x = 5, y = 10;
   printf("%d\n", x++ + ++y);  // 5 + 11 = 16
   printf("%d %d\n", x, y);    // 6 11
   return 0;
}
```

Q47: Predict Output

```
#include <stdio.h>
int main() {
   int arr[] = {1, 2, 3, 4, 5};
   printf("%d\n", *arr);       // 1
   printf("%d\n", *(arr + 2));  // 3
   printf("%d\n", arr[3]);     // 4
   return 0;
}
```

Q48: Predict Output

```
#include <stdio.h>
int main() {
   char str[] = "Hello";
   printf("%c\n", str[0]);     // H
```

```
    printf("%c\n", *(str + 1));  // e
    printf("%s\n", str + 2);     // llo
    return 0;
}
```

Tips for Exam:

1. Always check array bounds
2. Remember null terminator \0 in strings
3. Initialize variables before use
4. Use proper format specifiers
5. Practice writing code without IDE
6. Trace programs line by line
7. Remember operator precedence
8. Check loop conditions carefully

From <https://claude.ai/chat/fe937219-ed2e-4e6b-a9d6-53d97cb5332c>